Cheatography

D Scripting Toolbox Cheat Sheet

by Jesse K Phillips (JesseKPhillips) via cheatography.com/133808/cs/27440/

Read File

```
void main() {
   import std.file;
   auto data = readText("filename.ext");
}
```

readText reads the entire file into memory.

Matching Beginning or End

```
import std.algorithm;
auto data = "word word"
assert(data.startsWith("word"));
assert(data.endsWith("word"));
assert(data.skipOver("word"));
assert(data == " word");
```

skipOver will not only identify if the string is found at the beginning, but also move data past the string.

Removing Whitespace

```
import std.string;
auto data = " str ";
assert(data.strip == "str");
assert(data.stripLeft == "str");
assert(data.stripRight == "str");
```

strip Along with whitespace an argument can be past in to specify a specific string to strip off.

Find a String

```
import std.algorithm;
auto haystack = "some long name";
auto needle = "long";
if(!haystack.find(needle).empty)
    haystackHasNeedle(haystack);
```

find is a general function that works on any range. So it can be used with numbers and other containers. Similar to count and filter it is possible to specify more complex conditions.

Split Strings

Head and Tail

```
import std.range;
auto data = ["one", "two"];
auto head = data.front;
auto tail = data.dropOne;
```

dropOne modifies and returns the same range having removed the first element.

Slice

```
auto content = ["post" , "tweet", "video", "talk"];
auto begin = content[0..2];
auto end = content[2..$];
// content is ["post" , "tweet", "video", "talk"];
// begin is ['post', 'tweet']
// end is ['video', 'talk']
```

The right index is exclusive and does not include that indexes content. These made no modifications to the original array and no allocation was performed. This does mean modifying the slice will modify the original.

Remove the Middle of Array

```
import std.algorithm;
import std.typecons : tuple;
auto content = ["post" , "tweet", "video", "talk"];
content = content.remove(tuple(-1,3));
assert(content.equal(["post" , "talk"]) );
```

remove() resembles removing elements at a specified index. A tuple can be utilized to provide a sequence of indexes to be removed, and like slicing the first number is inclusive and last number is exclusive.

Execute a Program

```
import std.process;
import std.exception;
auto res = execute(["app name", "arg 1", "arg 2"]);
enforce(res.status == 0, "App exited with failure");
import std.stdio;
writeln(res.output);
```

execute will run the application to completion and store the output for review.

Handling Paths

```
import std.path;
buildNormalizedPath("foo/-bar/../peas"
            .absolutePath.expandTilde);
auto a = "foo";
auto b = "bar";
assert([a, b].joiner("/").equal(a.buildPath(b)));
assert("file.ext".extension == ".ext");
assert("file.ext".setExtension("gif") == "file.gif");
assert("file.ext".stripExtension == "file");
```

There are lots of utilities in std.path

```
import std.algorithm;
auto data = "one str\ntwo";
assert(data.splitter(" ").equal(
    ["one", "str\ntwo"]));
import std.array;
assert(data.split().equal(["-
one", "str", "two"]));
```

splitter is a lazy operation with no memory allocation, split eagerly evaluates into a new dynamic array. The second, when passing no arguments will also split on all whitespace including new lines.

By **Jesse K Phillips**
(JesseKPhillips)

Published 22nd May, 2021.
Last updated 22nd May, 2021.